Google

Building the Linux kernel with Clang

Linaro Connect ‘19

Signed-off-by: Tri Vo <trong@android.com>

Signed-off-by: Nick Desaulniers <ndesaulniers@google.com>

Project History

- 2009
  - LLVM META bug filed
- 2010
  - LLVM cfe-dev post about successful build+boot
- 2011
  - LLL Project
- 2012-2016
  - LLVMLinux Project
    - commits
    - archive
- 2016
  - Android Userspace 100% Clang (Google Pixel)
  - Pixel kernel was working at this time, but feature was punted to Pixel 2
- 2017
  - Google Pixel 2 ships first Clang built kernel
  - Clang patch stacks for LTS kernels (v4.4 and v4.9) and status update
  - Clang patch stacks for LTS kernels and status update
- 2018
  - ChromeOS starts shipping Clang built kernels
    - start
    - moved to default
    - Google Pixel 3 enables kernel LTO & CFI
- 2019 LLD, llvm-ar enabled, SCS

Status

- arm64 allyesconfig pretty much done (4 gcov patches in -next tree).
- arm32_v5, arm32_v6, arm32_v7, ppc32, ppc64le defconfigs build+boot out of the box.
  - A few arm32 configs have runtime issues we're working on.
- X86_64 requires **asm goto** (WIP) but otherwise runs well with 2 out of tree patches.

See cron jobs: https://travis-ci.com/ClangBuiltLinux/continuous-integration/builds

How Linux Improves LLVM

- Linux stresses Clang's support for extended inline assembly.
- Linux stresses Clang's integrated assembler (particularly pseudo-instructions).
- Linux stresses Clang's GNU C extension support.
- Linux stresses LLD's linker script and linker flag support.
- Linux boot time is **sensitive to code size**.
- Linux is (probably) the **largest** open source C codebase to throw at Clang.

Additions to LLVM

- `-fno-delete-null-pointer-checks` (landed in Clang 7)
- rN register naming for aarch64 (landed in Clang 7)
- `CONFIG_ARM64_LSE_ATOMICS` support (landed in Clang 8)
  - `-ffixed-x` # reserves general-purpose registers
  - `-fcall-saved-x` # specifies additional callee-saved registers
- **asm goto** (ETA clang 9)
- `__GCC_ASM_FLAG_OUTPUTS__` (landed in Clang 9)

Additions to Linux

- Support for Clang implementation of GCOV (ETA Linux 5.2)
- Fixes for various compiler warnings
  - `-Wsometimes-uninitialized`
  - `-Wenum-conversion`
  - `-Wpointer-bool-conversion`
  - `-Wsection`
  - `-Wignored-attributes`
  - `-Wheader-guard`
  - …

Fun bugs

Most recently (-Wsometimes-initialized):
https://github.com/ClangBuiltLinux/linux/issues/390
https://git.kernel.org/pub/scm/linux/kernel/git/stable/linux.git/tree/arch/powerpc/xmon/ppc-dis.c?h=v4.14.109#n167

```
const struct powerpc_opcode *opcode;
...
if (cpu_has_feature(CPU_FTRS_POWER9))
    dialect |= (PPC_OPCODE_POWER5 | PPC_OPCODE_POWER6 | PPC_OPCODE_POWER7
           | PPC_OPCODE_POWER8 | PPC_OPCODE_POWER9 | PPC_OPCODE_HTM
           | PPC_OPCODE_ALTIVEC | PPC_OPCODE_ALTIVEC2
           | PPC_OPCODE_VSX | PPC_OPCODE_VSX3),

  /* Get the major opcode of the insn.  */
  opcode = NULL;
  ...
  if (opcode == NULL)
    ...
```

Improvements to both sides

`git log` shows >1000 commit messages mentioning clang/llvm in the Linux kernel, and >100 in LLVM mentioning the Linux kernel.

KernelCI

- Currently building linux-next nightly on staging.kernelci.org.
- Moving to production ASAP, and potentially more branches.
- Linaro TCWG also testing builds of LLVM against Linux.

KernelCI

- Sends a concise build report
- Lets us quickly and easily triage issues
- Allows us to fix the problems **before they reach mainline**

next-clang/master build: 4 builds: 1 failed, 3 passed, 238 errors, 125 warnings (next-20190326)

kernelci.org bot <bot@kernelci.org>
to natechancellor, broonie, arnd.bergmann, ndesaulniers, kernel-build-reports, me, matthew.hart

next-clang/master build: 4 builds: 1 failed, 3 passed, 238 errors, 125 warnings (next-20190326)

Full Build Summary: https://staging.kernelci.org/build/next-clang/branch/master/kernel/next-20190326/

Tree: next-clang
Branch: master
Git Describe: next-20190326
Git Commit: ab8bba4ec4b9a1a63c02741e1e420d1d53d1a1c8
Git URL: http://git.kernel.org/pub/scm/linux/kernel/git/next/linux-next.git
Built: 1 unique architecture

Build Failure Detected:

arm64: clang version 8.0.0-svn353512-1~exp1~20190208132615.29 (branches/release_80)

allmodconfig: FAIL

Errors and Warnings Detected:

arm64: clang version 8.0.0-svn353512-1~exp1~20190208132615.29 (branches/release_80)

allmodconfig: 238 errors, 71 warnings
allnoconfig: 3 warnings
defconfig: 48 warnings
tinyconfig: 3 warnings

Errors summary:

1 tracing_map.c:(.text+0x6b18): undefined reference to `llvm_gcov_init'
1 tracing_map.c:(.text+0x6328): undefined reference to `llvm_gcda_start_file'
1 trace_uprobe.c:(.text+0xbaac): undefined reference to `llvm_gcov_init'
1 trace_uprobe.c:(.text+0xad34): undefined reference to `llvm_gcda_start_file'
1 trace_syscalls.c:(.text+0x4eb4): undefined reference to `llvm_gcov_init'
1 trace_syscalls.c:(.text+0x46a4): undefined reference to `llvm_gcda_start_file'
1 trace_stat.c:(.text+0x1ba8): undefined reference to `llvm_gcov_init'
1 trace_stat.c:(.text+0x181c): undefined reference to `llvm_gcda_start_file'
1 trace_stack.c:(.text+0x2774): undefined reference to `llvm_gcov_init'
1 trace_stack.c:(.text+0x238c): undefined reference to `llvm_gcda_start_file'
1 trace_seq.c:(.text+0x1f58): undefined reference to `llvm_gcov_init'
1 trace_seq.c:(.text+0x1d3c): undefined reference to `llvm_gcda_start_file'

What’s next?

- Finishing up being able to build various arch’s and configs. (build errors and warnings)
- Focus on adding new features to LLVM and polish.
  - **asm goto**
  - Improved stack slot reuse
  - Reduced warning false positives
  - Improved assembler support

Report bugs

- https://clangbuiltlinux.github.io/
- https://github.com/ClangBuiltLinux/linux/issues
  - We try to flag/save low hanging fruit for folks looking to get started contributing to Linux, see `good-first-issue` tag.
  - File a blocker: https://bugs.llvm.org/show_bug.cgi?id=4068 cc: ndesaulniers@google.com
- Compiler Explorer (godbolt.org): Great for sharing reproducers
- creduce: Minimize reproducers
- bear: compile_commands.json for compiler flags
- .<target>.o.cmd files produced by a build contain the exact compiler flags

Thanks!

Thanks to all of our contributors and folks working on this from both the LLVM and Linux side!

Thanks to Linaro for helping us test continuously, reporting and fixing bugs upstream!